CLAIMS

What is claimed is:

1. A method comprising:

determining whether a memory operation involves a misaligned memory address;

performing said memory operation with aligned memory accesses if said memory operation is determined as not involving a misaligned memory address; and

performing said memory operation with an algorithm including a merge operation and aligned memory accesses if said memory operation is determined as involving a misaligned memory address.

2. The method of claim 1 wherein said determining whether said memory operation involves a misaligned memory address comprises:

determining whether a source memory address for said memory operation is aligned with a memory boundary; and

determining whether a destination memory address for said memory operation is aligned with another memory boundary

3. The method of claim 2 wherein said performing said memory operation with said algorithm comprises:

processing first source data from a first portion of a source data block specified by said source memory address;

processing second source data from a second portion of said source data block;

determining whether enough unprocessed source data remains from said second portion to form a boundary aligned data block; and

processing third source data from a third portion of said source data block if said

determination indicates not enough unprocessed source data from said second portion remains.

4. The method of claim 3 wherein said processing said first source data comprises:

loading said first source data with an aligned memory load operation;

loading first unrelated data from a first portion of a destination space specified by said destination memory address with an aligned memory load operation;

merging said first source data with said first unrelated data to form a first merged data block; and

storing said first merged data block with an aligned memory store operation.

5. The method of claim 4 wherein said processing said first source data further comprises:

incrementing said source memory address to a first next boundary in said source data block; and

incrementing said destination memory address to a first next boundary in said destination space.

6. The method of claim 5 wherein said processing of said second source data comprises:

loading said second source data with an aligned memory load operation;

merging remaining first source data with said second source data to form a second merged data block having a length sufficient to fit between two consecutive boundaries; and

storing said second merged data block with an aligned memory store operation at said first next boundary in said destination space.

7. The method of claim 6 wherein said second source data is located at said first next boundary within said source data block.

8. The method of claim 6 wherein said processing of said second source data further comprises:

incrementing said source memory address to a second next boundary in said source data block; and

incrementing said destination memory address to a second next boundary in said destination space.

9. The method of claim 8 wherein said processing said third source data comprises:

loading said third source data with an aligned load operation;

merging remaining second source data with said third source data to form a remaining source data block;

loading second unrelated data from a second portion of said destination space with an aligned memory load operation;

merging said remaining source data block with said second unrelated data block to form a third merged data block; and

storing said third merged data block with an aligned memory store operation at said second next boundary in said destination space.

10. The method of claim 9 wherein said third source data is located at said second next boundary in said source data block.

11. The method of claim 1 wherein said memory operation is a memory copy.

12. The method of claim 1 wherein said memory operation is a memory move.

13. The method of claim 1 wherein said algorithm includes a packed merge

instruction to cause said merge operation.

14. The method of claim 13 wherein said merge operation comprises:

receiving a shift count of M;

shifting a first operand having a first set of L data elements left by 'L – M' data elements;

shifting a second operand having a second set of L data elements right by M data elements;

merging said shifted first set with said shifted second set to generate a resultant having L data elements.

15. The method of claim 14 wherein said merge operation is a variable merge operation wherein said shift count can be determined at runtime.

16. The method of claim 14 wherein said merge operation is a constant merge operation wherein said shift count is fixed at code compilation.

17. A method comprising:

evaluating a request for an unaligned memory operation having a source memory address and a destination memory address;

loading a first portion of source data from said source memory address for said unaligned memory operation with an aligned load operation;

loading a first boundary aligned destination data block including said destination memory address;

determining an amount of said first portion that can fit into said first boundary aligned destination data block starting at said destination memory address;

merging said first boundary aligned destination data block with said first portion

to form a first merged data block including said determined amount of said first portion; and

storing said first merged data block with an aligned store operation at an address of said first boundary aligned destination data block.

18. The method of claim 17 wherein said merging comprises:

receiving a shift count of M;

shifting a first operand having a first set of L data elements left by 'L – M' data elements;

shifting a second operand having a second set of L data elements right by M data elements;

merging said shifted first set with said shifted second set to generate a resultant having L data elements.

19. The method of claim 18 wherein said merge operation is a variable merge operation wherein said shift count can be determined at runtime.

20. The method of claim 18 further comprising:

loading a second portion of said source data with an aligned load operation;

determining whether any of said first portion of said source data has not been stored;

if said determination indicates any of said first portion has not been stored, merging any remaining unstored first portion source data with said second portion to form a second merged data block having a length sufficient to fit between two consecutive boundaries and storing said second merged data block to an address of a first boundary after said destination memory address with an aligned store operation;

and

if said determination indicates all of said first portion has been stored, storing said second portion of said source data to an address of said first boundary after said destination memory address with an aligned store operation.

21. The method of claim 18 further comprising:

loading a second portion of said source data with an aligned load operation;

determining whether said second portion includes sufficient data to form a boundary aligned data block;

if said determination indicates sufficient data is available in said second portion to form a boundary aligned data block, storing said second portion of said source data to an address of said first boundary after said destination memory address with an aligned store operation; and

if said determination indicates insufficient data is available in said second portion to form a boundary aligned data block, loading a second boundary aligned destination data block from said address of said first boundary after said destination memory address with an aligned store operation, merging said second portion with said second boundary aligned destination data block to form a second merged data block including entirety of said second portion and part of said second boundary aligned destination data block, and storing said second merged data block with an aligned store operation at said address of said first boundary after said destination memory address with an aligned store operation.

22. A machine readable medium having embodied thereon a computer program, said computer program being executable by a machine to perform a method comprising:

determining whether a memory operation involves a misaligned memory address;

performing said memory operation with aligned memory accesses if said memory operation is determined as not involving a misaligned memory address; and

performing said memory operation with an algorithm including a merge operation and aligned memory accesses if said memory operation is determined as involving a misaligned memory address.

23. The machine readable medium of claim 22 wherein said performing said memory operation with said algorithm comprises:

processing first source data from a first portion of a source data block specified by said source memory address;

processing second source data from a second portion of said source data block;

determining whether enough unprocessed source data remains from said second portion to form a boundary aligned data block; and

processing third source data from a third portion of said source data block if said determination indicates not enough unprocessed source data from said second portion remains.

24. The machine readable medium of claim 23 wherein said processing said first source data comprises:

loading said first source data with an aligned memory load operation;

loading first unrelated data from a first portion of a destination space specified by said destination memory address with an aligned memory load operation;

merging said first source data with said first unrelated data to form a first merged data block; and

storing said first merged data block with an aligned memory store operation.

25. The machine readable medium of claim 24 wherein said processing said first source data further comprises:

incrementing said source memory address to a first next boundary in said source data block; and

incrementing said destination memory address to a first next boundary in said destination space.

26. The machine readable medium of claim 25 wherein said processing of said second source data comprises:

loading said second source data with an aligned memory load operation;

merging remaining first source data with said second source data to form a second merged data block having a length sufficient to fit between two consecutive boundaries; and

storing said second merged data block with an aligned memory store operation at said first next boundary in said destination space.

27. The machine readable medium of claim 26 wherein said processing of said second source data further comprises:

incrementing said source memory address to a second next boundary in said source data block; and

incrementing said destination memory address to a second next boundary in said destination space.

28. The machine readable medium of claim 27 wherein said processing said third source data comprises:

loading said third source data with an aligned load operation;

merging remaining second source data with said third source data to form a remaining source data block;

loading second unrelated data from a second portion of said destination space with an aligned memory load operation;

merging said remaining source data block with said second unrelated data block to form a third merged data block; and

storing said third merged data block with an aligned memory store operation at said second next boundary in said destination space.

29. The machine readable medium of claim 22 wherein said merge operation comprises:

receiving a shift count of M;

shifting a first operand having a first set of L data elements left by 'L – M' data elements;

shifting a second operand having a second set of L data elements right by M data elements;

merging said shifted first set with said shifted second set to generate a resultant having L data elements.

30. The machine readable medium of claim 29 wherein said memory operation is a memory copy or a memory move.

31. The machine readable medium of claim 30 wherein said merge operation is a variable merge operation wherein said shift count can be determined at runtime.

32. The machine readable medium of claim 30 wherein said merge operation is a

constant merge operation wherein said shift count is fixed at code compilation.

33. An apparatus comprising:

an execution unit to execute an instruction requesting a memory operation, said instruction to cause said execution unit to:

determine whether a memory operation involves a misaligned memory address;

perform said memory operation with aligned memory accesses if said memory operation is determined as not involving a misaligned memory address; and

perform said memory operation with an algorithm including a merge operation and aligned memory accesses if said memory operation is determined as involving a misaligned memory address.

34. The apparatus of claim 33 wherein said perform said memory operation with said algorithm comprises:

processing first source data from a first portion of a source data block specified by said source memory address;

processing second source data from a second portion of said source data block;

determining whether enough unprocessed source data remains from said second portion to form a boundary aligned data block; and

processing third source data from a third portion of said source data block if said determination indicates not enough unprocessed source data from said second portion remains.

35. The apparatus of claim 34 wherein said processing said first source data

comprises:

loading said first source data with an aligned memory load operation;

loading first unrelated data from a first portion of a destination space specified by said destination memory address with an aligned memory load operation;

merging said first source data with said first unrelated data to form a first merged data block; and

storing said first merged data block with an aligned memory store operation.

36. The apparatus of claim 35 wherein said processing said first source data further comprises:

incrementing said source memory address to a first next boundary in said source data block; and

incrementing said destination memory address to a first next boundary in said destination space.

37. The apparatus of claim 36 wherein said processing of said second source data comprises:

loading said second source data with an aligned memory load operation;

merging remaining first source data with said second source data to form a second merged data block having a length sufficient to fit between two consecutive boundaries; and

storing said second merged data block with an aligned memory store operation at said first next boundary in said destination space.

38. The apparatus of claim 37 wherein said processing of said second source data further comprises:

incrementing said source memory address to a second next boundary in said source data block; and

incrementing said destination memory address to a second next boundary in said destination space.

39. The apparatus of claim 38 wherein said processing said third source data comprises:

loading said third source data with an aligned load operation;

merging remaining second source data with said third source data to form a remaining source data block;

loading second unrelated data from a second portion of said destination space with an aligned memory load operation;

merging said remaining source data block with said second unrelated data block to form a third merged data block; and

storing said third merged data block with an aligned memory store operation at said second next boundary in said destination space.

40. The apparatus of claim 39 wherein said merge operation comprises:

receiving a shift count of M;

shifting a first operand having a first set of L data elements left by 'L – M' data elements;

shifting a second operand having a second set of L data elements right by M data elements;

merging said shifted first set with said shifted second set to generate a resultant having L data elements.

41. The apparatus of claim 40 wherein said memory operation is a memory copy or a memory move.

42. The apparatus of claim 41 wherein said merge operation is a variable merge operation wherein said shift count can be determined at runtime.

43. The apparatus of claim 41 wherein said merge operation is a constant merge operation wherein said shift count is fixed at code compilation.

44. A system comprising:

a memory to store data; and

a processor coupled to said memory on a bus, said processor operable to process a request for a memory operation, said processor comprising:

a bus unit to receive an instruction requesting a memory operation; and

an execution unit coupled to said bus unit, said execution unit to execute said instruction requesting said memory operation, said instruction to cause said execution unit to:

determine whether a memory operation involves a misaligned memory address;

perform said memory operation with aligned memory accesses if said memory operation is determined as not involving a misaligned memory address; and

perform said memory operation with an algorithm including a merge operation and aligned memory accesses if said memory operation is determined as involving a misaligned memory address.

45. The system of claim 44 wherein said perform said memory operation with said

algorithm comprises:

processing first source data from a first portion of a source data block specified by said source memory address;

processing second source data from a second portion of said source data block;

determining whether enough unprocessed source data remains from said second portion to form a boundary aligned data block; and

processing third source data from a third portion of said source data block if said determination indicates not enough unprocessed source data from said second portion remains.

46. The system of claim 45 wherein said processing said first source data comprises:

loading said first source data with an aligned memory load operation;

loading first unrelated data from a first portion of a destination space specified by said destination memory address with an aligned memory load operation;

merging said first source data with said first unrelated data to form a first merged data block;

storing said first merged data block with an aligned memory store operation;

incrementing said source memory address to a first next boundary in said source data block; and

incrementing said destination memory address to a first next boundary in said destination space.

47. The system of claim 46 wherein said processing of said second source data comprises:

loading said second source data with an aligned memory load operation;

merging remaining first source data with said second source data to form a second merged data block having a length sufficient to fit between two consecutive boundaries;

storing said second merged data block with an aligned memory store operation at said first next boundary in said destination space;

incrementing said source memory address to a second next boundary in said source data block; and

incrementing said destination memory address to a second next boundary in said destination space.

48. The system of claim 47 wherein said processing said third source data comprises:

loading said third source data with an aligned load operation;

merging remaining second source data with said third source data to form a remaining source data block;

loading second unrelated data from a second portion of said destination space with an aligned memory load operation;

merging said remaining source data block with said second unrelated data block to form a third merged data block; and

storing said third merged data block with an aligned memory store operation at said second next boundary in said destination space.

49. The system of claim 48 wherein said merge operation comprises:

receiving a shift count of M;

shifting a first operand having a first set of L data elements left by 'L – M' data elements;

shifting a second operand having a second set of L data elements right by M data elements;

merging said shifted first set with said shifted second set to generate a resultant having L data elements.

50. The system of claim 49 wherein said memory operation is a memory copy or a memory move.

51. The system of claim 50 wherein said merge operation is a variable merge operation wherein said shift count can be determined at runtime.

52. The system of claim 50 wherein said merge operation is a constant merge operation wherein said shift count is fixed at code compilation.